

# Crowd Pedestrian Counting Considering Network Flow Constraints in Videos <sup>1</sup>

Liqing Gao, Yanzhang Wang and Xin Ye  
 Institute of Information and Decision Technology  
 Dalian University of Technology,  
 Dalian 116023, P.R. China  
 Jian Wang  
 Center for Combinatorics, LPMC-TJKLC  
 Nankai University,  
 Tianjin 300071, P.R. China

## Abstract

A quadratic programming method with network flow constraints is proposed to improve crowd pedestrian counting in video surveillance. Most of the existing approaches estimate the number of pedestrians within one frame, which result in inconsistent predictions in temporal domain. In this paper, firstly, we segment the foreground of each frame into different groups, each of which contains several pedestrians. Then we train a regression-based map from low level features of each group to its person number. Secondly, we construct a directed graph to simulate people flow, whose vertices represent groups of each frame and edges represent people moving from one group to another. Then, the people flow can be viewed as an integer flow in the constructed directed graph. Finally, by solving a quadratic programming problem with network flow constraints in the directed graph, we obtain a consistent pedestrian counting. The experimental results show that our method can improve the crowd counting accuracy significantly.

## Index Terms

crowd pedestrian counting, network flow constraints, quadratic programming model, linear programming model

## I. INTRODUCTION

Crowd pedestrian counting is a challenging problem in computer vision, due to heavy long-term occlusion and various perspective distortion in different environments. It has gained significant interest in areas such as public security, resource management and public transportation monitoring. There are three types of counting techniques including counting by detection, counting by statistics and counting by tracking.

People can be detected by a pedestrian detector [1], face detector and head-shoulder detector [2]. In the pedestrian detection approach, a binary classifier is trained using common features, such as haar wavelets and histogram of oriented gradients (HOG) [3]. Then the trained classifier can be applied to search for pedestrians by sliding window in image pyramid. The detection performance can be further improved by deformable parts model [4]. Pedestrian detection is scaling and distortion insensitive due to pyramid window searching and deformable parts model, which lead to cross-scene counting techniques. However, despite of remarkable progression, the accuracy of counting by detection heavily suffers from high missing rates of detectors, especially in high occlusion level.

Statistic counting methods are regression methods [5]–[9], [13], [21] by training a map from low level features to the number of people directly. Antoni B. Chan and Nuno Vasconcelos [5]–[7] utilize gaussian process regression method and bayesian poisson regression method to obtain the correspondence between the features (segment features, edges features and texture features) and crowd number. D.Conte et al [8], [9] applied support vector regression method to learn the map from salient points based features to crowd number.

Although the training procedure needs some elaborate work, including feature selection, these methods are more robust and efficient than pedestrian detection methods in high-density crowd scene. Therefore they gain extensive popularity in crowd counting. There are other machine learning methods applied to crowd counting, such as sparse representation [15] and deep learning [19]. However, almost all statistic methods use only one frame to predict pedestrian number, which result in inconsistent predictions. It means that these methods may output different values for the same group of people in consecutive frames.

Recent tracking methods view people flow as a network flow and view a pedestrian as a continuous trajectory, which can be modeled as a 1-flow (or a path) in the network, then utilize network flow methods to multi-object tracking tasks. Anton Milan et al [10] model the problem as a multi-label conditional random field on network, and find a set of continuous trajectory by  $\alpha$ -expansion algorithm. Horesh Ben Shitrit et al [16] formulate tracking multiple people whose walking paths may intersect as a multi-commodity network flow problem. As multi-object tracking tasks need to identify every single person, time-independent people detector are used to find out possible people sites per frame and then linked these sites into consistent trajectories by global optimization on networks. Thus, this method may success in the scene that pedestrians are sparse. However, it is hard to deal with ID switches in crowd scene.

Counting in crowd scene is much different to multi-object tracking task, because we are only concerned with the total number of pedestrians per frame. In crowd scene, people are clustered into several groups and we can only track each group as a whole. Thus, instead of modeling each possible single-person position in each frame as a vertex, we model each group as a vertex of a network (or a directed graph), and a consistent counting result should satisfy the network flow constraints on the network. For each group of pedestrians, the regression based method is used to predict its person number, which may not be consistent with other frames. Then we obtain final solutions satisfying network flow constraints by solving a mathematical programming problem.

In this paper, we utilize background subtraction method to obtain foregrounds of fixed scene videos. The foregrounds are supposed to be all pedestrians and we cluster the foreground of each frame to several groups. Then we train a support vector regressor mapping from extracted features of each group to its person number. Later, we construct a quadratic programming model and a linear programming model to improve the predicted people number of group. Finally, by summing up number of all groups in same frame, we obtain the final counting results. It is worth noting that our quadratic and linear programming method can also be applied to any regression counting approaches that segment foreground firstly, such as methods in [7], [9]. The main components of proposed method are depicted in Fig. 1.

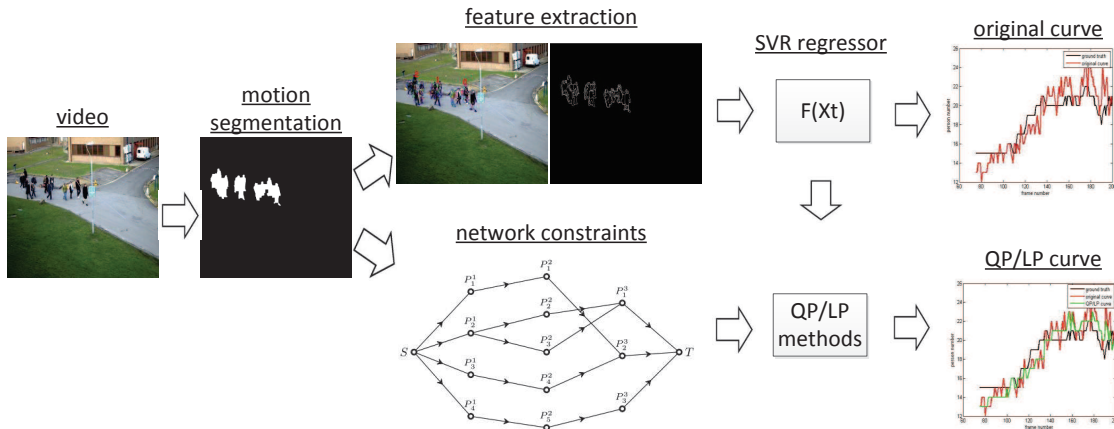


Fig. 1. The main components of our method

The experimental results show that the proposed method outperforms Conte's method in [9], and our quadratic programming approach improves the accuracy significantly and performs a consistent crowd counting in videos. We further compare quadratic programming approach with linear programming one.

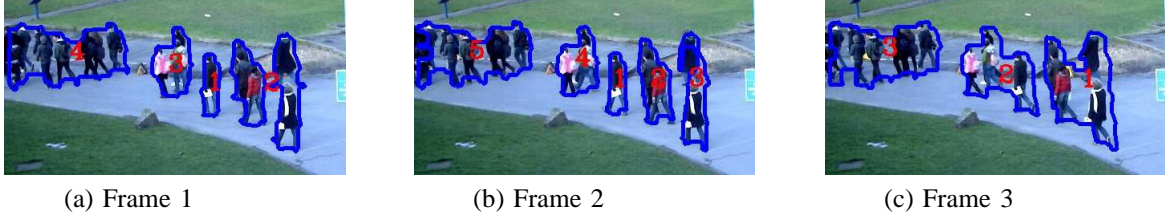


Fig. 2. Three consecutive frames and their segmentations.

The experiments show that quadratic programming approach performs better than linear programming one in most cases.

The paper is organized as follows. Section II introduces the foreground segmentation method and SVR regression counting method used for experiments in this paper. In Section III, we structure the network flow model and introduce the network flow constraints for crowd pedestrian counting. Section IV-A and Section IV-B adopt the quadratic programming model and linear programming model to solve the network flow constraints problem, respectively. Experimental results of different methods to the crowd counting problem are presented in Section V. Finally, we give some concluding remarks in Section VI.

## II. FOREGROUND SEGMENTATION AND REGRESSION COUNTING METHOD

In order to verify the effectiveness of network flow constraints, we propose a simple regression counting method. Given a video of specific scene, we first use frame differential method to obtain the foreground image of each frame. Then we further process the foreground image by erosion and dilation operations. Suppose all moving objects in the video are pedestrians, then each connected region in foreground image is viewed as a group. As shown in Fig.2(a), the pedestrians are clustered into four groups.

Let  $C$  be a group in the segmented image. We convert  $C$  into a feature vector, then feed it into a regressor. The output of the regressor is the estimated number of persons in the group. Let  $T$  be the vector  $(x_c, y_c, w, h, \psi, \ell, \zeta, \theta)$  and the number of people  $n_C$  in  $C$  can be estimated by

$$n_C = f(T) \quad (1)$$

where

- $(x_c, y_c)$  is the center of gravity of group  $C$ .
- $(w, h)$  is the width and height of bounding rectangle of group  $C$ .
- $\psi$  is total number of pixels of group  $C$ .
- $\ell$  is perimeter of group  $C$ .
- $\zeta$  is total number of edge pixels contained in group  $C$  detected by Canny edge detector.
- $\theta$  is total number of SURF feature points contained in group  $C$ .

Support vector machine has a strong learning capability that is proposed for classical dichotomy problems. We utilize support vector regression algorithm to train a regressor. For testing, we use the trained regressor to estimate the number of individuals in each group. It should be noticed that the proposed regression method requires scene dependent training. For a new camera setup, the learning process should be repeated.

## III. NETWORK FLOW CONSTRAINTS FOR CROWD COUNTING

Let  $(I_1, I_2, \dots, I_n)$  be a sequence of frames in a video. For each image  $I_i$ , we segment the foreground of  $I_i$  into  $m_i$  groups  $P_1^i, P_2^i, \dots, P_{m_i}^i$  and each group is a connected region in frame  $i$ . Let  $P^i = \{P_1^i, P_2^i, \dots, P_{m_i}^i\}$  and  $P = \bigcup_{i=1}^n P^i$ . For a fixed scene, regions that people entering and exiting are unchangeable, which are often close to the image boundary. We denote region from which people entering the fixed scene with  $S$  and region through which people exiting the fixed scene with  $T$ . An example is shown in Fig.3. The zone  $S$  and the zone  $T$  are the same region, because they both allow pedestrian entering and exiting in these videos.



Fig. 3. Entering region  $S$  and exiting region  $T$  in the specific scene.

**Definition III.1.** Let  $D(V, A)$  be a directed graph, whose vertex set is  $V = P \cup \{S, T\}$  and the arc set is set  $A$  such that

- (1)  $\langle P_i^t, P_j^{t+1} \rangle \in A$  if and only if  $P_i^t$  overlaps  $P_j^{t+1}$ ;
- (2)  $\langle S, P_i^t \rangle \in A$  if  $P_i^t$  overlaps  $S$ , and  $\langle P_i^t, T \rangle \in A$  if  $P_i^t$  overlaps  $T$ ;
- (3)  $\langle S, P_i^t \rangle \in A$  if frame  $t$  is the first frame of image sequence, and  $\langle P_i^t, T \rangle \in A$  if frame  $t$  is the last frame of image sequence;
- (4)  $\langle S, P_i^t \rangle \in A$  if  $P_i^t \cap P_k^{t-1} = \emptyset$  for any  $P_k^{t-1} \in P^{t-1}$ , and  $\langle P_i^t, T \rangle \in A$  if  $P_i^t \cap P_j^{t+1} = \emptyset$  for any  $P_j^{t+1} \in P^{t+1}$ .

Clearly, directed graph  $D$  is a network with source  $S$  and sink  $T$ . Since frame number is increasing along the direction of each arc, then network  $D$  is acyclic. For simplicity, we identify groups with vertices of  $D$  in this paper. It should be noticed that foreground segmentation may not be as accurate as possible. When group tracking fails, it may cause that some  $P_i^t$  has no intersections with any groups in  $P^{t-1}$  or some  $P_i^t$  has no intersections with any groups in  $P^{t+1}$ . If  $P_i^t$  has no intersections with any groups in  $P^{t-1}$ , then  $\langle S, P_i^t \rangle$  is an arc according to (4) of definition III.1. If  $P_i^t$  has no intersections with any groups in  $P^{t+1}$ , then  $\langle P_i^t, T \rangle$  is an arc according to (4) of definition III.1. As we will see later, it makes our method more robust.

Since it is impossible to construct the network of whole video in real time. For any frame  $I_c$ , we use the set of  $(2\ell + 1)$  frames  $\{I_{c-\ell}, I_{c-\ell+1}, \dots, I_{c-1}, I_c, I_{c+1}, \dots, I_{c+\ell}\}$  to construct the network. We call the graph as  $(2\ell + 1)$ -layer network centered at frame  $c$ , denoted by  $H(c, \ell)$ . Fig. 2 shows three consecutive frames. The corresponding 3-layer network centered at frame 2, denoted by  $H(2, 1)$ , is shown in Fig. 4.

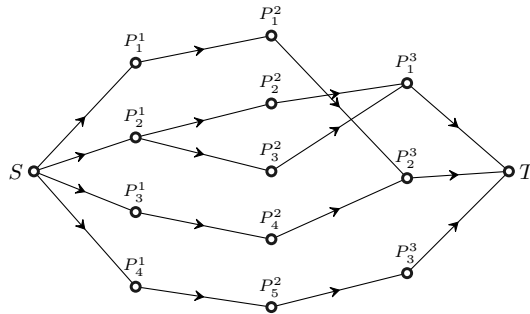


Fig. 4. 3-layer network  $H(2, 1)$  corresponding to three frames in Fig. 2.

For digraph  $D$ , the subgraph induced by  $V_1 \subset V$  is denoted by  $D[V_1]$ . Let  $D' = D[V \setminus \{S, T\}]$ , we can decompose  $D'$  into  $p$  weakly connected components  $D'_1, D'_2, \dots, D'_p$ . Then for each  $i \in \{1, 2, \dots, p\}$ , induced digraph  $D[V(D'_i) \cup \{S, T\}]$  forms a network with source  $S$  and sink  $T$ . We call them weakly connected sub-networks  $D_i$ . As an example, three weakly connected sub-networks corresponding to network  $H(2, 1)$  in Fig 4 are shown in Fig 5.

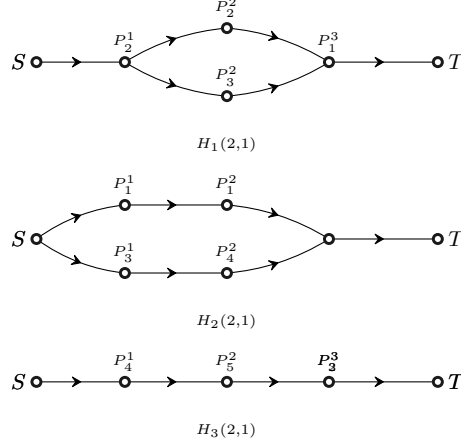


Fig. 5. Connected component decomposition of Networks.

Suppose that the segmentation of foreground never divide a single person into two or more groups. Then the actual number of people in each group is an integer value. Define  $f$  be a function

$$f : A \mapsto \mathbb{N}. \quad (2)$$

such that

- For any arc  $\langle P_i^t, P_j^{t+1} \rangle \in A$ , define  $f(\langle P_i^t, P_j^{t+1} \rangle)$  be the actual person number moving from group  $i$  at frame  $t$  to group  $j$  at frame  $t + 1$ ;
- For any arc  $\langle S, P_i^t \rangle \in A$ , define  $f(\langle S, P_i^t \rangle)$  be the actual person number of group  $i$  at frame  $t$ ;
- For any arc  $\langle P_i^t, T \rangle \in A$ , define  $f(\langle P_i^t, T \rangle)$  be the actual person number of group  $i$  at frame  $t$ .

Let

$$\begin{aligned} f^+(P_i^t) &= \sum_{\langle P_i^t, y \rangle \in A} f(\langle P_i^t, y \rangle), \\ f^-(P_i^t) &= \sum_{\langle x, P_i^t \rangle \in A} f(\langle x, P_i^t \rangle). \end{aligned} \quad (3)$$

Clearly,  $f^+(P_i^t) = f^-(P_i^t)$  and they both represent the actual person number in group  $i$  at frame  $t$ , we set  $f(P_i^t) = f^+(P_i^t) = f^-(P_i^t)$ . Moreover, people only enter the scene from region  $S$  and exit the scene through region  $T$ , which implies that  $f^+(S) = f^-(T)$ . Thus, function  $f$  on arc set  $A$  is an integer  $(S, T)$ -flow in network  $D$ . The network flow constraints of network  $D$  can be defined as follows.

$$\begin{cases} \sum_{\langle x, P_i^t \rangle \in A} f(\langle x, P_i^t \rangle) = \sum_{\langle P_i^t, y \rangle \in A} f(\langle P_i^t, y \rangle) = f(P_i^t), & \text{for } P_i^t \in V \setminus \{S, T\} \\ \sum_{\langle S, P_j^t \rangle \in A} f(\langle S, P_j^t \rangle) = \sum_{\langle P_k^t, T \rangle \in A} f(\langle P_k^t, T \rangle) \end{cases} \quad (4)$$

Therefore, when we get the predicted number of pedestrians of  $(2\ell + 1)$  frames in a video, a consistent result should satisfy the network flow constraints of network  $D$ .

#### IV. MATHEMATICAL PROGRAMMING MODEL

The regression method described in Section II can predict the number of people for each group in the network. We denote the predicted value of group  $i$  at frame  $t$  by  $\hat{f}(P_i^t)$ . Since the prediction is processed per frame, it can never guarantee consistent results, which means it will violate the network flow constraints of network  $D$ . In order to obtain better pedestrian crowd counting results, we put forward an integer quadratic programming model and an integer linear programming model on network  $D$  to improve the predictions of regression methods, respectively.

##### A. Quadratic Programming Model

The quadratic programming model on network  $D$  is constructed as follows:

$$\begin{aligned}
 \min \quad & \sum_{P_i^t \in V \setminus \{S, T\}} w(\hat{f}(P_i^t)) \cdot (f(P_i^t) - \hat{f}(P_i^t))^2 \\
 \text{s.t.} \quad & f(P_i^t) = \sum_{\langle \mathbf{x}, P_i^t \rangle \in A} f(\langle \mathbf{x}, P_i^t \rangle), \quad \text{for } P_i^t \in V \setminus \{S, T\} \\
 & f(P_i^t) = \sum_{\langle P_i^t, \mathbf{y} \rangle \in A} f(\langle P_i^t, \mathbf{y} \rangle), \quad \text{for } P_i^t \in V \setminus \{S, T\} \\
 & \sum_{\langle S, P_j^t \rangle \in A} f(\langle S, P_j^t \rangle) = \sum_{\langle P_k^t, T \rangle \in A} f(\langle P_k^t, T \rangle)
 \end{aligned} \tag{5}$$

Where  $f(P_i^t)$ 's and  $f(\langle P_i^t, P_j^{t+1} \rangle)$ 's are integer valuables,  $w(\hat{f}(P_i^t))$  represents the reliability of prediction  $\hat{f}(P_i^t)$  to group  $i$  at frame  $t$ .

1) *Model Solution:* It is difficult to obtain the solution of proposed model. However, by relaxing the integer valuables  $f(P_i^t)$ 's,  $f(\langle P_i^t, P_j^{t+1} \rangle)$ 's into real valuables, the model change to be a quadratic programming problem with only linear equation constraints. Since the objective function is strongly convex, the problem has unique optimal solution. The lagrange function of this model can be written as follow.

$$\begin{aligned}
 L(f, \lambda^-, \lambda^+, \mu) = & \sum_{P_i^t \in V \setminus \{S, T\}} w(\hat{f}(P_i^t)) \cdot (f(P_i^t) - \hat{f}(P_i^t))^2 \\
 & + \sum_{P_i^t \in V \setminus \{S, T\}} \lambda_{P_i^t}^- \left( f(P_i^t) - \sum_{\langle \mathbf{x}, P_i^t \rangle \in A} f(\langle \mathbf{x}, P_i^t \rangle) \right) \\
 & + \sum_{P_i^t \in V \setminus \{S, T\}} \lambda_{P_i^t}^+ \left( f(P_i^t) - \sum_{\langle P_i^t, \mathbf{y} \rangle \in A} f(\langle P_i^t, \mathbf{y} \rangle) \right) \\
 & + \mu \left( \sum_{\langle S, P_j^t \rangle \in A} f(\langle S, P_j^t \rangle) - \sum_{\langle P_k^t, T \rangle \in A} f(\langle P_k^t, T \rangle) \right).
 \end{aligned} \tag{6}$$

In mathematical optimization, the Karush-Kuhn-Tucker (KKT) conditions are first order necessary conditions for a solution in nonlinear programming to be optimal, provided that some regularity conditions are satisfied. For convex programming problems, KKT conditions are also sufficient ones. The KKT condition



of proposed problem can be derived as follows.

$$\left\{ \begin{array}{l} \sum_{\langle \mathbf{x}, P_i^t \rangle \in A} \mathbf{f}(\langle \mathbf{x}, P_i^t \rangle) = \mathbf{f}(P_i^t), \text{ for } P_i^t \in V \setminus \{S, T\}; \\ \sum_{\langle P_i^t, \mathbf{y} \rangle \in A} \mathbf{f}(\langle P_i^t, \mathbf{y} \rangle) = \mathbf{f}(P_i^t), \text{ for } P_i^t \in V \setminus \{S, T\}; \\ \sum_{\langle S, P_j^t \rangle \in A} \mathbf{f}(\langle S, P_j^t \rangle) = \sum_{\langle P_k^t, T \rangle \in A} \mathbf{f}(\langle P_k^t, T \rangle); \\ \lambda_{P_j^t}^+ + \lambda_{P_k^{t+1}}^- = 0, \text{ for } \langle P_j^t, P_k^{t+1} \rangle \in A; \\ -\mu + \lambda_{P_j^t}^- = 0, \text{ for } \langle S, P_j^t \rangle \in A; \\ \lambda_{P_k^t}^+ + \mu = 0, \text{ for } \langle P_k^t, T \rangle \in A; \\ \sum_{P_i^t \in V \setminus \{S, T\}} \mathbf{w}(\hat{\mathbf{f}}(P_i^t)) \cdot (\mathbf{f}(P_i^t) - \hat{\mathbf{f}}(P_i^t)) + \lambda_{P_i^t}^- + \lambda_{P_i^t}^+ = 0, \text{ for any } P_i^t \in V \setminus \{S, T\}. \end{array} \right. \quad (7)$$

Suppose  $|V| = n$  and  $|A| = m$ , the KKT condition is a linear system with  $3n + m - 5$  variables and  $3n + m - 5$  equations. Therefore, we can solve the quadratic programming problem by solving KKT linear system.

Recall that network  $D$  can be decomposed into several weakly connected sub-networks  $D_1, D_2, \dots, D_p$ . Since each sub-network is independent, then we can divide the original problem into some simple sub problems. That means we only need to solve the quadratic programming model on each weakly connected sub-network  $D_i$ , for  $i = 1, 2, \dots, p$ .

If some weakly connected sub-networks of  $D$  is a directed path with  $n$  internal vertices like  $H_3(2, 1)$  shown in Fig 5 (b), network flow constraints turn out to be  $\mathbf{f}(P_{l_1}^1) = \mathbf{f}(P_{l_2}^2) = \dots = \mathbf{f}(P_{l_n}^n)$ . So we can set them all be  $f$ , and the optimal problem becomes as follow.

$$\min \sum_{t=1}^n \mathbf{w}(\hat{\mathbf{f}}(P_{l_t}^t)) \cdot (f - \hat{\mathbf{f}}(P_{l_t}^t))^2. \quad (8)$$

Finally we get the consistent prediction

$$f = \frac{\sum_{t=1}^n \mathbf{w}(\hat{\mathbf{f}}(P_{l_t}^t)) \cdot \hat{\mathbf{f}}(P_{l_t}^t)}{\sum_{t=1}^n \mathbf{w}(\hat{\mathbf{f}}(P_{l_t}^t))}. \quad (9)$$

2) *Reduction of Lagrangian Multiplies:* In order to simplify the KKT linear system, we analyze the constraints on lagrangian multiplies. These equations are independent with other valuables, and each of them is related to an arc of graph  $D$ .

$$\left\{ \begin{array}{l} \lambda_{P_j^t}^+ + \lambda_{P_k^{t+1}}^- = 0, \text{ for } \langle P_j^t, P_k^{t+1} \rangle \in A; \\ -\mu + \lambda_{P_j^t}^- = 0, \text{ for } \langle S, P_j^t \rangle \in A; \\ \lambda_{P_k^t}^+ + \mu = 0, \text{ for } \langle P_k^t, T \rangle \in A. \end{array} \right. \quad (10)$$

Let  $\lambda_S^+ = -\mu$  and  $\lambda_T^- = \mu$ , and set  $\bar{A}$  be the union set of  $A$  and  $\{\langle S, T \rangle\}$ . Then equations (10) can be unified as equation (11).

$$\lambda_x^+ + \lambda_y^- = 0, \text{ for any } \langle \mathbf{x}, \mathbf{y} \rangle \in \bar{A}. \quad (11)$$

For any arc  $\langle \mathbf{x}, \mathbf{y} \rangle \in \bar{A}$ , lagrangian multiplier  $\lambda_x^+ = -a$  can be viewed as out-weight of arc  $\langle \mathbf{x}, \mathbf{y} \rangle$ ,  $\lambda_y^- = a$  can be viewed as in-weight of  $\langle \mathbf{x}, \mathbf{y} \rangle$ . Fig 6 shows an edge representation of two lagrangian multipliers. Define  $Deg^+(\mathbf{x}) = \{\mathbf{y} | \langle \mathbf{x}, \mathbf{y} \rangle \in \bar{A}\}$  and  $Deg^-(\mathbf{x}) = \{\mathbf{y} | \langle \mathbf{y}, \mathbf{x} \rangle \in \bar{A}\}$ . Clearly, equation (11) implies  $\lambda_x^- = \lambda_y^-$  for any  $\mathbf{x}, \mathbf{y} \in Deg^+(\mathbf{z})$  and  $\lambda_x^+ = \lambda_y^+$  for any  $\mathbf{x}, \mathbf{y} \in Deg^-(\mathbf{z})$ .



Fig. 6. (I) The arc representation of lagrangian multipliers. (II) Reduction of lagrangian multipliers.

**Definition IV.1.** For any two arcs  $\langle u, v \rangle \in \bar{A}$  and  $\langle x, y \rangle \in \bar{A}$ , define relation  $R$  on arc set  $\bar{A}$  as  $\langle u, v \rangle R \langle x, y \rangle$  if and only if  $u = x$  or  $v = y$ .

According to the defined relation  $R$ , it is easy to prove that the transitive closure of  $R$  is an equivalence relation. So arc set  $A$  can be partitioned into distinct  $m$  equivalent classes, say  $A_1, A_2, \dots, A_m$ . For example, the equivalent classes of  $\bar{A}(H_1) = A(H_1) \cup \{\langle S, T \rangle\}$  in Figure 5 are sets  $A_1(H_1)$ ,  $A_2(H_1)$  and  $A_3(H_1)$ , where

$$\begin{aligned} A_1(H_1) &= \{\langle S, T \rangle, \langle S, P_2^1 \rangle, \langle P_1^3, T \rangle\}; \\ A_2(H_1) &= \{\langle P_2^1, P_2^2 \rangle, \langle P_2^1, P_3^2 \rangle\}; \\ A_3(H_1) &= \{\langle P_2^2, P_1^3 \rangle, \langle P_3^2, P_1^3 \rangle\}. \end{aligned}$$

Arcs in the same equivalent class have the same out-weights and in-weights, and they shared one common valuable. Therefore  $2|V| - 4$  lagrangian multipliers are reduced to  $m$  valuables. The arc representation of reduced lagrangian multipliers of  $H_1$ ,  $H_2$  and  $H_3$  are given in Fig 7, and all the lagrangian multipliers are labeled on the arcs of the network and arcs with same label are equivalent. As shown in Fig 7, arcs in  $A_1(H_1)$  share one common lagrangian multiplier  $\mu$ . Arcs in  $A_2(H_1)$  share common lagrangian multiplier  $a$  and arcs in  $A_3(H_1)$  share one common lagrangian multiplier  $b$ .

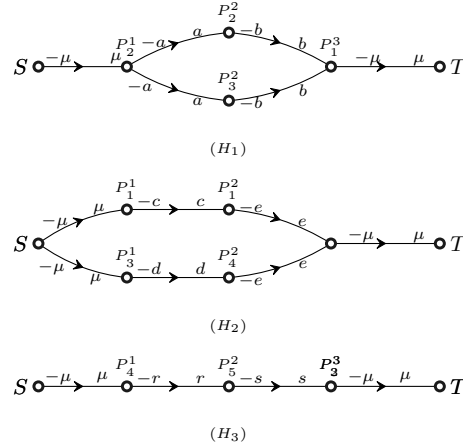


Fig. 7. Equivalence classes of lagrangian multipliers.

### B. Linear Programming Model

The network flow constraints problem can also be solved by a linear programming model. Instead of minimum mean squared error, we use minimum mean absolute error as objective function. The model is



constructed as follows:

$$\begin{aligned}
\min \quad & \sum_{P_i^t \in V \setminus \{S, T\}} \mathbf{w}(\hat{\mathbf{f}}(P_i^t)) \cdot |\mathbf{f}(P_i^t) - \hat{\mathbf{f}}(P_i^t)| \\
s.t. \quad & \mathbf{f}(P_i^t) = \sum_{\langle \mathbf{x}, P_i^t \rangle \in A} \mathbf{f}(\langle \mathbf{x}, P_i^t \rangle), \quad \text{for } P_i^t \in V \setminus \{S, T\}; \\
& \mathbf{f}(P_i^t) = \sum_{\langle P_i^t, \mathbf{y} \rangle \in A} \mathbf{f}(\langle P_i^t, \mathbf{y} \rangle), \quad \text{for } P_i^t \in V \setminus \{S, T\}; \\
& \sum_{\langle S, P_j^t \rangle \in A} \mathbf{f}(\langle S, P_j^t \rangle) = \sum_{\langle P_k^t, T \rangle \in A} \mathbf{f}(\langle P_k^t, T \rangle).
\end{aligned}$$

In order to further simplify the model, let  $\mathbf{h}(P_i^t) \geq |\mathbf{f}(P_i^t) - \hat{\mathbf{f}}(P_i^t)|$ , the model can be changed into the following model, which can be solved by linear programming method.

$$\begin{aligned}
\min \quad & \sum_{P_i^t \in V \setminus \{S, T\}} \mathbf{w}(\hat{\mathbf{f}}(P_i^t)) \mathbf{h}(P_i^t) \\
s.t. \quad & \mathbf{f}(P_i^t) - \mathbf{h}(P_i^t) - \hat{\mathbf{f}}(P_i^t) \leq 0, \quad \text{for any } P_i^t \in V \setminus \{S, T\}; \\
& \hat{\mathbf{f}}(P_i^t) - \mathbf{h}(P_i^t) - \mathbf{f}(P_i^t) \leq 0, \quad \text{for any } i \in V \setminus \{S, T\}; \\
& \mathbf{f}(P_i^t) = \sum_{\langle \mathbf{x}, P_i^t \rangle \in A} \mathbf{f}(\langle \mathbf{x}, P_i^t \rangle), \quad \text{for } P_i^t \in V \setminus \{S, T\}; \\
& \mathbf{f}(P_i^t) = \sum_{\langle P_i^t, \mathbf{y} \rangle \in A} \mathbf{f}(\langle P_i^t, \mathbf{y} \rangle), \quad \text{for } P_i^t \in V \setminus \{S, T\}; \\
& \sum_{\langle S, P_j^t \rangle \in A} \mathbf{f}(\langle S, P_j^t \rangle) = \sum_{\langle P_k^t, T \rangle \in A} \mathbf{f}(\langle P_k^t, T \rangle).
\end{aligned}$$

### C. QPLm and LPLm Algorithms

To sum up, we propose two algorithms according to two different models. One is SVR regression method with quadratic programming on  $m$ -layer networks, we call it QPLm method. The other one is SVR regression method with linear programming on  $m$ -layer networks and we call it LPLm method. The details of the algorithms are depicted in Algorithm 1.

In order to determine  $\mathbf{w}(\hat{\mathbf{f}}(P_i^t))$ 's in quadratic programming model or linear programming model, we further analyze the trained regressor in Section II. Suppose the training set is denoted by  $\{(P_i^t, \mathbf{g}(P_i^t)) : 1 \leq i \leq m_t, t \in I\}$ , where  $I$  is the set of selected frame numbers. where  $m_t$  is number of groups at frame  $t$  and  $\mathbf{g}(P_i^t)$  is the actual number of people in group  $P_i^t$ . Let  $\hat{\mathbf{f}}(P_i^t)$  denotes the number of people of each group  $P_i^t$  predicted by the trained regressor. Then the training data can be represented by set  $\{(P_i^t, \mathbf{g}(P_i^t), \hat{\mathbf{f}}(P_i^t)) : 1 \leq i \leq m_t, t \in I\}$ . For a given predicted value  $\tau$ , there must be a list of groups whose person number are predicted as  $\tau$ , that is, set  $G(\tau) = \{\mathbf{g}(P_i^t) : \hat{\mathbf{f}}(P_i^t) = \tau, 1 \leq i \leq m_t, t \in I\}$ . We calculate the mean and variance of  $G(\tau)$ . Finally, we regulate the regressor by subtracting mean of  $G(\tau)$  when the predicted value equals  $\tau$ , and let weight  $\mathbf{w}(\tau)$  be the variance of set  $G(\tau)$ .

## V. EXPERIMENTAL RESULTS AND ANALYSIS

We implemented the proposed method by using Visual Studio 2010 with Opencv2.4.8 on Windows 7, and utilize PETS2009 datasets [24] to assess the performance of the proposed method. The PETS 2009

**Input:** A sequence of frames  $I_1, I_2, \dots, I_n$ .  
**Output:** Crowd pedestrian numbers  $f_1, f_2, \dots, f_n$  according to each frame.

```

1 for  $j = 1; j \leq n; j++$  do
2   for  $t = j - 2\ell; t \leq j + 2\ell; t++$  do
3     if  $t \geq 1 \& \& t \leq n$  then
4       Segment  $I_t$  into  $m_t$  groups and let  $P^k = \{P_1^t, P_2^t, \dots, P_{m_t}^t\}$ ;
5       for  $i = 1; i \leq m_t; i++$  do
6         Utilize trained regressor to predict person number  $\hat{f}(P_i^t)$  in group  $P_i^t$ ;
7       end
8     end
9   else
10    Let  $m_t = 0$  and  $P^t = \emptyset$ ;
11  end
12 end
13 Construct network  $H(j, \ell)$  with vertex set  $V(j, \ell) = \left(\bigcup_{t=j-2\ell}^{j+2\ell} P^t\right) \cup \{S, T\}$ ;
14 Decompose network  $H(j, \ell)$  into  $p$  weakly connected sub-networks  $H_1(j, \ell), H_2(j, \ell), \dots, H_p(j, \ell)$ ;
15 for  $i = 1; i \leq p; i++$  do
16   Obtain  $S, T$ -flow  $\mathbf{f}$  on  $H_i(j, \ell)$  by solving quadratic programming model or linear programming model on  $H_i(j, \ell)$ ;
17 end
18  $f_j = 0$ ;
19 for  $i = 1; i \leq m_j; i++$  do
20    $f_j = f_j + \mathbf{f}(P_i^j)$ ;
21 end
22 end
23 return  $f_1, f_2, \dots, f_n$ ;

```

**Algorithm 1:** QPL( $2\ell + 1$ ) method/LPL( $2\ell + 1$ ) method

dataset is organized in four sections, but our attention is mainly focused on the section S1 that was used to benchmark algorithms for the ‘‘Person Count and Density Estimation’’ PETS2009 and 2010 contests. The experimental videos involve two different views captured by using two cameras that contemporaneously acquired the same scene from different points of view. We used eight videos of this dataset, namely S1.L1.13-57, S1.L1.13-59, S1.L2.14-06 and S1.L3.14-17 in view 1, and S1.L1.13-57, S1.L2.14-06, S1.L2.14-31 and S3.MF.12-43 in view 2. For short, S1.L1.13-57(1), S1.L1.13-59(1), S1.L2.14-06(1), S1.L3.14-17(1), S1.L1.13-57(2), S1.L2.14-06(2), S1.L2.14-31(2), and S3.MF.12-43(2).

In order to use the proposed method for crowd counting, we had first to train the support vector regressor (SVR). The minimum size of the training set needed to achieve an acceptable performance, as the statistical learning theory by Vapnik and Chervonenkis has demonstrated, depends on both the complexity of the problem and the complexity of the estimator. The training set was constructed by manually collecting some frames from the video. In this paper, we select 30-40 frames from each video for training, and the rest of each is for testing. The test has been carried out by comparing actual number of people and the number of people calculated by different algorithms. The indices used to report the performance are the Mean Absolute Error (MAE) and the Mean Relative Error (MRE) defined as follows,

$$MAE = \frac{1}{N} \cdot \sum_{i=1}^N |G(i) - T(i)|, \quad (12)$$

$$MRE = \frac{1}{N} \cdot \sum_{i=1}^N \frac{|G(i) - T(i)|}{T(i)}, \quad (13)$$

where  $N$  is the number of frames of the test video and  $G(i)$  and  $T(i)$  are the guessed and the true number

of persons in the  $i$ -th frame, respectively.

TABLE I  
COUNTING ESTIMATION ERROR OF ORIGINAL METHOD, METHODS WITH NETWORK CONSTRAINTS SOLVED BY QPL $m$  AND LPL $m$  ON CONSIDERED DATASET.

Method	S1.L1.13-57(1)		S1.L1.13-59(1)		S1.L2.14-06(1)		S1.L3.14-17(1)	
	MAE	MRE	MAE	MRE	MAE	MRE	MAE	MRE
original	1.29	5.58%	1.14	7.29%	4.77	17.35%	<b>2.82</b>	<b>11.68%</b>
LPL3	1.25	5.47%	1.07	7.16%	4.74	17.29%	2.82	11.68%
LPL7	1.26	5.51%	0.97	6.28%	4.76	17.33%	2.92	11.94%
LPL11	1.25	5.45%	0.94	6.39%	4.74	17.27%	2.92	11.94%
LPL15	1.25	5.51%	0.89	5.94%	4.71	17.22%	2.92	11.94%
LPL19	1.27	5.60%	0.85	5.25%	4.69	17.17%	2.92	11.94%
LPL23	1.29	5.64%	<b>0.83</b>	<b>5.08%</b>	4.65	17.06%	2.92	11.94%
QPL3	1.15	5.06%	1.07	7.10%	4.74	17.29%	2.82	11.68%
QPL7	1.21	5.27%	1.01	6.31%	4.76	17.33%	2.88	11.84%
QPL11	1.23	5.36%	1.02	6.51%	4.74	17.27%	2.88	11.84%
QPL15	1.18	5.21%	0.96	6.11%	4.71	17.22%	2.88	11.84%
QPL19	1.14	5.08%	0.93	5.73%	4.69	17.17%	2.88	11.84%
QPL23	<b>1.13</b>	<b>5.04%</b>	0.96	5.90%	<b>4.65</b>	<b>17.06%</b>	2.88	11.84%

Method	S1.L1.13-57(2)		S1.L2.14-06(2)		S1.L2.14-31(2)		S3.MF.12-43(2)	
	MAE	MRE	MAE	MRE	MAE	MRE	MAE	MRE
original	8.53	24.79%	10.54	38.61%	2.97	9.57%	0.49	9.99%
LPL3	8.30	24.04%	10.61	38.67%	2.88	9.47%	0.32	7.43%
LPL7	8.28	23.94%	10.31	37.81%	2.88	9.44%	0.21	3.85%
LPL11	8.25	23.79%	10.19	37.35%	2.95	9.65%	0.16	3.11%
LPL15	8.27	23.86%	10.04	36.63%	2.94	9.61%	0.17	3.72%
LPL19	8.32	24.04%	9.90	35.89%	3.00	9.77%	0.25	5.27%
LPL23	8.44	24.32%	9.68	34.74%	3.03	9.86%	0.16	3.19%
QPL3	8.16	23.73%	10.46	38.32%	2.88	9.47%	0.40	8.52%
QPL7	8.10	23.48%	10.10	37.30%	<b>2.84</b>	<b>9.31%</b>	0.28	5.13%
QPL11	8.05	23.29%	9.90	36.65%	2.93	9.56%	0.22	4.13%
QPL15	8.04	23.30%	9.70	35.80%	2.90	9.48%	0.21	3.86%
QPL19	<b>8.02</b>	<b>23.20%</b>	9.52	34.98%	2.94	9.60%	0.19	3.86%
QPL23	8.03	23.20%	<b>9.25</b>	<b>33.69 %</b>	2.99	9.73%	<b>0.11</b>	<b>2.20%</b>

TABLE II  
COUNTING ESTIMATION ERROR OF PROPOSED METHOD AND CONTE'S ON THE CONSIDERED DATASET.

Method	Conte [9]		QPL23	
	MAE	MRE	MAE	MRE
S1.L1.13-57(1)	1.36	6.80%	<b>1.13</b>	<b>5.04%</b>
S1.L1.13-59(1)	2.55	16.30%	<b>0.96</b>	<b>5.90%</b>
S1.L2.14-06(1)	5.40	20.80%	<b>4.65</b>	<b>17.06%</b>
S1.L3.14-17(1)	<b>2.81</b>	<b>15.10%</b>	2.88	11.84%
S1.L1.13-57(2)	<b>4.45</b>	<b>15.10%</b>	8.03	23.20%
S1.L2.14-06(2)	12.17	30.70%	<b>9.25</b>	<b>33.69%</b>
S1.L2.14-31(2)	7.55	23.60%	<b>2.99</b>	<b>9.73%</b>
S3.MF.12-43(2)	3.26	10.88%	<b>0.11</b>	<b>2.20%</b>

On these datasets, we implement the crowd counting method based on SVR regression algorithm firstly, which is named as original method. Then we have performed two groups of tests. The tests in the first are aimed at analyzing the experimental results of QPL $m$  method. The second tests are focus on analyzing the experimental results of LPL $m$  method. The details of the algorithm are depicted in Algorithm 1. The experimental results are shown in Table I, and the curves of crowd estimation of different methods in four videos are shown in Fig.8.

From the results reported in Table I, we can see that the error of our proposed method tends to descending with the network layer increasing. Combining with Fig.8, we find that the errors of SVR predictor from all frames of video S1.L1.13-57(1) (Fig.8.1) fluctuate smoothly. Its errors are gaussian-like noises, which are suited to solve by QP method. Thus, QP method performs well. However, in the original curve of video S1.L1.13-59(1) (Fig.8.2) there are several sharp jumps. As L1 norm deals with laplacian-like noise well, LP method produces a better result than QP method on video S1.L1.13-59(1). Since errors of trained predictors

are often gaussian-like, QP method is better than LP method in most cases.

Comparing the original curve with the ground truth curve of each figure, the curve of original method oscillates a lot as it uses only one frame to predict the number of persons. While our methods can smooth out the oscillations and predict more precisely. The proposed methods are highly effective to reduce the errors when there are nontrivial network flow constraints. Otherwise, it will have no improvement. Such as in Fig. 8.4, there is no remarkable promotion between our original method and the QP method. One of the reasons is that there is only one group in most frames of these videos, and the only group either connect to vertex  $S$  or connect to vertex  $T$  such that no nontrivial network flow constraints are formed in the network. Another reason is that our algorithm of foreground segmentation is not robust enough such that network flow constraint fails. Thus, the proposed methods have no improvement on video S1.L3.14-17(1).

Table II presents the comparison between the counting accuracy of our proposed method and that of Conte's [9] method. It is worth noting that our method has a significant performance improvement comparing to Conte's result, except video S1.L3.14-17(1) and video S1.L1.13-57(2). We find that extracted features of these two videos can not well adapt to the rapid changes of the density of crowd. In the future, we will focus on improving the features of groups of videos.

## VI. CONCLUSION

In this paper, we propose the network flow constraints for crowd pedestrian counting at first time, then put forward an integer quadratic programming model and an integer linear programming model to improve the predictions of regression methods. For simplicity, integer variables are relaxed into real ones. Then integer quadratic programming models can be solved by solving linear equations and integer linear programming models can be solved by simplex methods. According to the comparison results between counting results of QPL $m$  algorithm and that of LPL $m$  algorithm, for fixed regression algorithm, it is clear that the method with network constraints solved by the QPL $m$  algorithm obtain better results than that by LPL $m$  algorithm in most videos. Our method improve counting results significantly and have lower error than Conte's method in most test videos. In the future, we shall try some better foreground segment algorithm and other better features to improve the performance. And we shall also consider the original integer programming problem directly, since the relaxation of integer variables leads to only approximate solutions.

## REFERENCES

- [1] Khatoon, Robina, Syed Muhammad Saqlain, and Stamatia Bibi. A robust and enhanced approach for human detection in crowd. Multitopic Conference (INMIC), 2012 15th International. IEEE, 2012.
- [2] Li M, Zhang Z, Huang K, et al. Estimating the number of people in crowded scenes by mid based foreground segmentation and head-shoulder detection[C]. Pattern Recognition, 2008. ICPR 2008. 19th International Conference on. IEEE, 2008: 1-4.
- [3] Dalal, Navneet, and Bill Triggs. Histograms of oriented gradients for human detection. Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. Vol. 1. IEEE, 2005.
- [4] Felzenszwalb, Pedro F., et al. Object detection with discriminatively trained part-based models. Pattern Analysis and Machine Intelligence, IEEE Transactions on 32.9 (2010): 1627-1645.
- [5] Chan A B, Vasconcelos N. Modeling, clustering, and segmenting video with mixtures of dynamic textures[J]. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2008, 30(5): 909-926.
- [6] Chan A B, Liang Z S J, Vasconcelos N. Privacy preserving crowd monitoring: Counting people without people models or tracking[C]. Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on. IEEE, 2008: 1-7.
- [7] Chan A B, Vasconcelos N. Counting people with low-level features and Bayesian regression[J]. Image Processing, IEEE Transactions on, 2012, 21(4): 2160-2177.
- [8] Conte D, Foggia P, Percannella G, et al. Counting moving people in videos by salient points detection[C]. Pattern Recognition (ICPR), 2010 20th International Conference on. IEEE, 2010: 1743-1746.
- [9] Conte D, Foggia P, Percannella G, et al. Counting moving persons in crowded scenes[J]. Machine vision and applications, 2013, 24(5): 1029-1042.
- [10] Milan A, Leal-Taix L, Schindler K, et al. Joint tracking and segmentation of multiple targets[C]. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015: 5397-5406.
- [11] Ge W, Collins R T. Marked point processes for crowd counting[C]. Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on. IEEE, 2009: 2913-2920.
- [12] Arteta C, Lempitsky V, Noble J A, et al. Interactive object counting[M]. Computer VisionCECCV 2014. Springer International Publishing, 2014: 504-518.

- [13] Chen K, Gong S, Xiang T, et al. Cumulative attribute space for age and crowd density estimation[C]. Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on. IEEE, 2013: 2467-2474.
- [14] Tang S, Andriluka M, Schiele B. Detection and tracking of occluded people[J]. International Journal of Computer Vision, 2014, 110(1): 58-69.
- [15] Foroughi H, Ray N, Zhang H. Robust people counting using sparse representation and random projection[J]. Pattern Recognition, 2015.
- [16] Ben Shitrit H, Berclaz J, Fleuret F, et al. Multi-commodity network flow for tracking multiple people[J]. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2014, 36(8): 1614-1627.
- [17] Rittscher J, Tu P H, Krahnstoever N. Simultaneous estimation of segmentation and shape[C]. Computer Vision and Pattern Recognition (CVPR), 2005 IEEE Computer Society Conference on. IEEE, 2005, 2: 486-493.
- [18] Yu Z, Gong C, Yang J, et al. Pedestrian counting based on spatial and temporal analysis[C]. Image Processing (ICIP), 2014 IEEE International Conference on. IEEE, 2014: 2432-2436.
- [19] Zhang C, Li H, Wang X, et al. Cross-scene crowd counting via deep convolutional neural networks[C]. Proc. CVPR. 2015.
- [20] Bondi E, Seidenari L, Bagdanov A D, et al. Real-time people counting from depth imagery of crowded environments[C]. Advanced Video and Signal Based Surveillance (AVSS), 2014 11th IEEE International Conference on. IEEE, 2014: 337-342.
- [21] Loy C C, Gong S, Xiang T. From semi-supervised to transfer counting of crowds[C]. Computer Vision (ICCV), 2013 IEEE International Conference on. IEEE, 2013: 2256-2263.
- [22] Zhao T, Nevatia R, Wu B. Segmentation and tracking of multiple humans in crowded environments[J]. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2008, 30(7): 1198-1211.
- [23] Albiol A, Silla M J, Albiol A, et al. Video analysis using corner motion statistics[C]. Proceedings of the IEEE International Workshop on Performance Evaluation of Tracking and Surveillance. 2009: 31-38.
- [24] PETS: <http://www.cvg.rdg.ac.uk/PETS2009/> (2009).

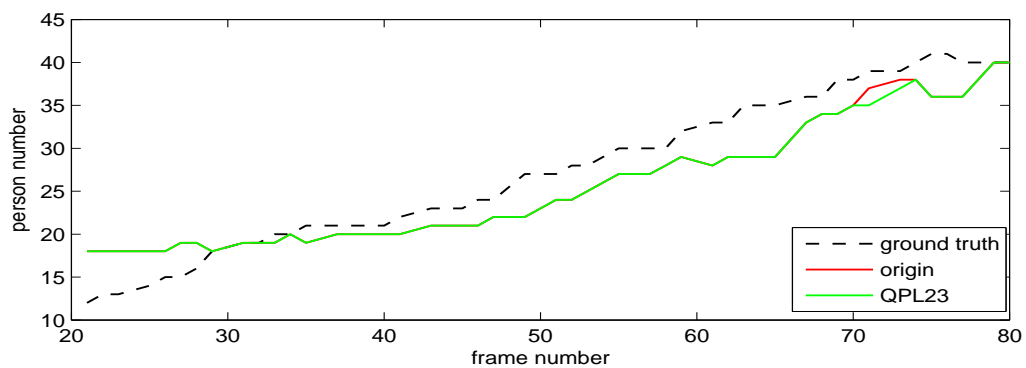
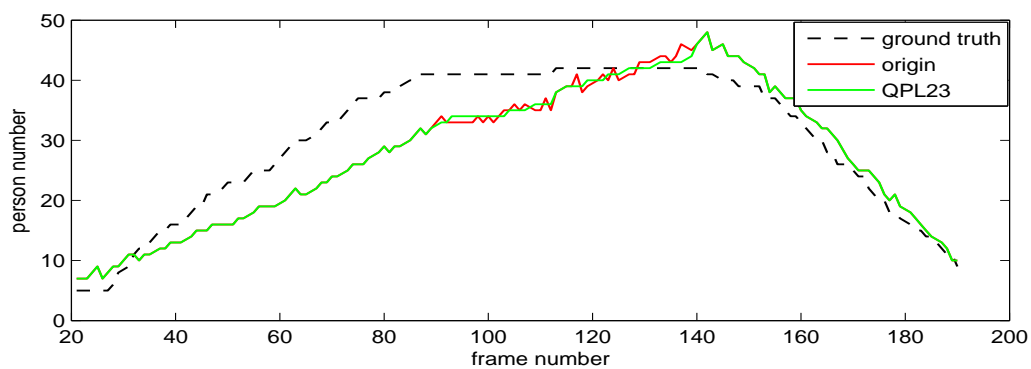
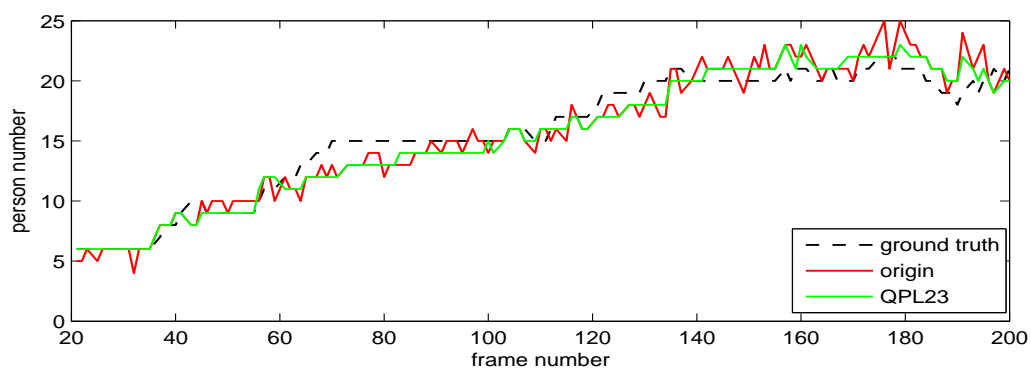
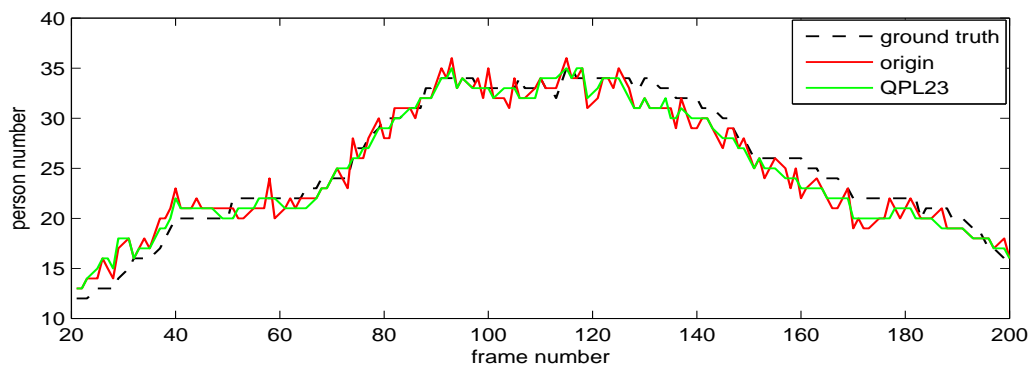


Fig. 8. Curves of person number estimated by different methods and the ground truth. x-axis presents frame number.